WHAT IS CLAIMED IS:

Claim 1:

1. A method of handling software locks in a multiple instruction processor computing system wherein software locks inquiries are handled on two tracks, a first conventional software lock handling process and a second process for handling communal software locks, wherein said second process handles inquiries from inquirers and comprises:

i. determining by a communal lock processor associated with a particular mid-level cache whether a lock inquiry from an inquirer is for a Communal SoftWare Lock (CSWL) or for a conventional software lock,

ii. processing said CSWL lock inquiry if said determination determines that the inquiry is for a CSWL, else allowing a conventional software lock process to process said conventional software lock request.

Claim 2:

2. A method of handling software locks as set forth in claim 1 wherein step (i) further comprises:

(a) determining, by a mid-level cache, if said lock request is from another mid-level cache through a side door, and if so, processing said lock request as a CSWL lock request, or

(b) determining if said lock request is from an instruction processor associated with said mid-level cache, and if so, determining if said lock request is for a CSWL by reference to a mapping of all available CSWLs, and if so, performing step (ii).

Claim 3:

3. The method of claim 2 wherein step (b) further comprises:

determining whether an inquiry is a request for a lock or a status report, and if a status report, sending information on the status of the lock to the inquirer which in such case is a local instruction processor.

Claim 4:

4. The method of claim 1 wherein said step (ii) comprises determining if the CSWL is mapped to the mid-level cache receiving said inquiry, and if not, sending a request for said CSWL to a mapped mid-level cache owning said CSWL, but if the CSWL is mapped to the mid-level cache receiving said inquiry, determining if the CSWL is present in said mid-level cache, and if not, requesting a cache line for said CSWL.

Claim 5:

5. The method of claim 4 wherein said requesting of a cache line for said CSWL is made through a communication channel used for ordinary, non-CSWL data.

Claim 6:

6. The method of claim 4 wherein if said CSWL is present in said mid-level cache, making a determination whether a requested lock value passed in the CSWL inquiry is a same value as an extant value which is in said CSWL located in said mid-level cache.

Claim 7:

7. The method of claim 5 wherein if said requested lock value and said extant lock value are the same, updating the extant value to a passed new value and preparing a status successful value, but if the extant value and the requested lock value are not the same, preparing an unsuccessful status value, and in both events, sending said prepared status value to said inquirer.

Claim 8:

8. The method of claim 5 wherein if said CSWL is located in said mid-level cache, making a determination whether a requested CSWL is available or locked to a previous inquirer, and if available, locking said CSWL and preparing a value to pass indicating said lock was locked to said inquirer making said request, but if said CSWL is not available, preparing a value to pass indicating said lock was not available to said inquirer making said request.

Claim 9:

9. The method of claim 1 further comprising;

establishing a mapping of all CSWLs for a partition before start-up of said partition,

setting up a set of memory registers containing addresses of said all CSWLs for said partition.

Claim 10:

10. The method of claim 9 wherein each CSWL is mapped to a particular mid-level cache.

Claim 11:

11. The method of claim 1 further comprising:

establishing a mapping of all CSWLs for a computer system before start-up of said computer system,

setting up a set of memory registers containing addresses of said all CSWLs for said computer system.

Claim 12:

A method of handling software locks in a multiple instruction processor computing system wherein software locks inquiries are handled on two tracks, a first conventional software lock handling process and a second process for handling communal software locks, comprising:

prior to running an operating system in said multiple instruction processor computing system,

determining which software locks will be high contention locks,

assigning each high contention lock to a particular one of a set of mid-level caches in said multiple instruction processor system wherein said high contention locks are communal locks,

setting up a system for holding said assignments so as to enable a determining by each of said particular ones of said set of mid-level caches which of said particular ones is assigned to each said high contention lock,

running said operating system and allowing application programs to run, using either or both of said first conventional software lock handling process and said second process for

handling said communal software locks as needed by said operating system and or application programs.

Claim 13:

13. A method for handling communal software locks in a multiprocessor computer system having a set of mid-level caches with a radial connection among said mid-level caches for transferring signals related to said communal software locks and wherein said communal software locks (CSWLs) may be set or not set, wherein said processing comprises setting or reporting on a set condition by sending signals to a requesting processor, and wherein said setting or reporting is done in and by a one of said mid-level caches.

Claim 14:

14. The method of claim 13 wherein a mapping assigns each CSWL to a particular one of said mid-level caches, and wherein each CSWL request checks said mapping so as to direct said each CSWL request to an appropriate one of said particular ones of said mid-level caches, that appropriate one being an assigned mid-level cache for a CSWL which is a subject of said CSWL request.

Claim 15:

15. The method of claim 14 wherein if a mid-level cache does not have present a requested CSWL at a time when it receives a CSWL request for a CSWL to which it is assigned by said mapping, said mid-level cache requests a cache line for said requested CSWL to which it is assigned by said mapping, so as to obtain the said CSWL and then perform said processing on said requested CSWL for a requesting processor making a request for said CSWL.